

SBACE

GAZETTE

Volume: 2

Issue: 3 MAY/JUNE, 1983

FROM THE EDITOR'S DESK

The deadline for the July/August newsletter will be August 2nd.

The material and opinions in this Gazette are those of the individual authors and do not necessarily reflect the opinions of the South Bay Atari Computer Enthusiasts. The material in this Gazette may be copied by any other User Group providing credit is given to the authors.

Please address all Correspondence to:

James A. Jengo
SBACE
5025 Range Horse Lane
Rolling Hills Est., Calif. 90274

Monthly meetings of S.B.A.C.E. are held on the 3rd Thursday of each month at 7:00 PM at:

HW Computers
2301 Artesia Blvd.
Redondo Beach, Calif. 90278

Our esteemed officers are:

President	Gerald Bransford
Vice President	Alan Haskell
Secretary/Treasurer	James Jengo
Librarian	Harry Koons
Newsletter Editor	Sue Whitehead

SBACE GAZETTE is in no way affiliated with ATARI, Inc. ATARI is a trademark of Atari, Inc.

Please don't forget to check the Newsletter Books in the library every month for Newsletters sent to us (on an exchange basis) by other Groups! I have listed below a sampling of the wide variety of information available:

- CURRENT NOTES: a clever way to insert multiple printer control codes (using Letter Perfect) using the search and replace command instead of manually typing in multiple control codes, multiple times (by David Geller).

How to input an alternate character set using the NEC printer (by Jack McKirgan II).

AD ASTRA is the bimonthly journal of the Atari Micro-computer Network. The Network is for ham radio operators. They have over 250 members and want more. They have internationally attended meetings weekly! For more info:

The Atari Microcomputer Network
Jack McKirgan II, WD8BNG
4749 S.R. 207 N.E.
Washington C.H., Ohio 43160
(phone: 614-869-3597)

Peeks & Pokes (by Becky Hanneman):

POKE 16,64:poke 53774,64 disables the BREAK
key until change graphics mode or Sys Reset.

When LOADING or SAVING hit CTRL 2 after RETURN
and the console speaker will beep when the
operation is done.

PEEK 53775 checks to see if any key is pressed
at the time of the PEEK. The value returned is
251 if a key is pressed, 255 if no key pressed.

POKE 766,1 prints CTRL characters (like the cursor
movement arrows) on the screen instead of per-
forming their functions.

PEEK 53770 returns a random integer between 0-255.

POKE 16,255 disables the keyboard.

POKE 702,n has the effect of pressing the CAPS/
LOWR key: n=0 for lower case, n=64 for caps,
n=128 for CTRL characters.

POKE 694,128 gives inverse video (0 for normal).

- CONTROL

POKE 580,1 Cold Start the system (like turning
the power switch off & on (by Mike Aldrich).

- CIO

An interesting article on the C/65 compiler from OSS.

-CONTACT

If you would like to look at the variable table while in the middle of a program without saving it disk, type SAVE "E:" and press RETURN. The computer will start printing your program on the screen and it will start with the variable tables. If you press BREAK quickly, you will have the desired screen listing.

If you press CTRL 2 the console buzzer will sound. Try pressing it while an operation is in progress (e.g. a disk copy). The keystroke will be stored in a buffer which will be read when the operation is over causing the buzzer to sound, waking you up (by Lorraine Green).

A list of Atari modifications to the 400/800/810.

POKE 838,166:POKE 839,238 cause everything to be sent to a printer which would normally be printed on the screen. POKE 838,128:POKE 839,246 gets back to normal.

-KEEPING P.A.C.E.

An article on how to use the DOS Copy function to enter program code into a disk file without using the BASIC or Assembler cartridges (e.g. if you can't find them but want to program) by Shane Rolin.

To do a COLDSTART from DOS choose option M and enter address F125 or E477 (this is just like turning the power off and on); for a WARMSTART (like SYS RESET) enter address F128 or E474 (Shane Rolin).

An article on how to simply install a switch on the 810 drive to allow you to ignore write protect tape on a diskette (by Bob Haden).

An article on interfacing the computer to the telephone (e.g. for dialing numbers) through the joystick ports (includes schematics & program) by Paul Serotta.

-Atari Computer Assoc. of Orange County

An article on bubble and shell sorts by Jim Budelman.

DOS 2.0 Binary File Load written in FORTH (Greg Beauton).

An article on how to add audio to a cassette program without erasing the program while adding the audio (Chuck Hosick).

-A B A C U S

CLIST & HUNT utilities for string searches written in fig-FORTH (by G. Smith & J. Peters).

An article on using TV artifacting in Gr. 0 & 8 (KC ACE).

Listing of JONESTRM program (by Frank C. Jones) and quite a few BBS phone numbers.

A new database program FILE FAX combines most of the good features of Filemanager 800 & Data Perfect plus adding a HELP function and the ability to handle large data bases comprising more than one diskette!!!

A tutorial on using random numbers in ways you may not have thought of (by Antonio Leal).

-KEEPING P.A.C.E.

Directions on how to modify an "el-chep-o" transistor radio to work from the computer's audio output, thus enabling audio with a monitor (Ed J. Reynolds WB3ERE).

A bunch of cute articles: e.g. how to double the speed of the 810 by doubling the input line voltage; technical update on new Erase Only Memory; the Black Hole Diode.

The BYLAWS of Pittsburgh ACE, Inc.

-ATARI COMPUTER ASSOC. OF ORANGE COUNTY

Some useful XIO (DOS) commands (by Josh Pine):

RENAME	XIO 32,#1,0,0,"D:TEMP.CAROL"
DELETE	XIO 33,#1,0,0,"D:TEMP.BAS"
LOCK FILE	XIO 35,#1,0,0,"D:TEMP.BAS"
UNLOCK FILE	XIO 36,#1,0,0,"D:TEMP.BAS"
FORMAT DISK	XIO 254,#1,0,0,"D1:"

-PAGE 6

A program to create text windows on Gr. 9,10 or 11 screens

-ATARI COMPUTER CLUB OF OKC, INC.

An article (by Andee White) on binary & hex number systems

The Macrotronics Atari Screen Printer Interface (a review)
by

James A. Jengo

If you do much (or even not so much) graphics work, you have probably wished you could print the screen display on your printer. In fact, I'm sure many of you have with programs such as the one by Computer Age Software (which works very nicely, by the way, but with one exception). A problem arises however if you have a screen display comprised of a non-standard display list (i.e. not a version of Graphics Modes 0-8). Certain commercial programs and of course many custom-made programs will use "home-made" display lists to intermix text and graphics in effective and novel ways. Unfortunately, the otherwise very effective program by Computer Age Software will not allow "dumping" (what a horrible sounding word!) displays from custom display lists.

Fortunately, the Macrotronics Atari Screen Printer Interface solves this problem, and in a GRAND way. It not only will print out ANY type of display (with standard or custom display lists), but also will print out any Player/Missile Graphics characters that happen to be on the screen (if you wish), and if all that wasn't enough, it does all this WITHOUT the 850 Interface Module!!!

Macrotronics supplies a long ribbon cable with a combination double joystick plug/interface board which simply and unobtrusively plugs into joystick ports 3&4, with the other end plugging into your printer (specify type: Centronics 739, Trendcom 200, IDS 440G or 445G, Epson MX70, MX80 or MX100 (with Graphtrax Plus), Okidata (with Okigraph) or NEC 8023A).

The documentation is well written, thorough and easy to understand with multiple examples. And, most importantly, it all works like it says it's supposed to! One simply boots up the system with the supplied autoboot disk (it doesn't matter if a cartridge is in place since this works under cartridge or machine control); enter the type of printer you have (numeric choice); remove the disk and go on with whatever you wanted to do. When you want to "dump" the screen you just press CTRL P and your program will stop executing, the screen dump will occur, and then your original program will continue on. Alternatively, one may start the print-out under program control using a USR statement in BASIC (or a JSR in ASSEMBLY).

Multiple options are available. The aspect ratio can easily be changed allowing either changes in the horizontal and/or vertical scaling. Additionally, one can reverse the colors to get white characters on a black background. Also, if the gray scale representation of the screen's colors is not pleasing, you don't have to change the computer's color registers; instead, you can specify that the "Screen Printer's" color registers be used: these can be changed without altering the computer program/display in any way. You also have the option to have your print-out scaled either according to the hue OR the luminance of either the computer's or the "Screen Printer's" color registers. One can also print data which has been "fine scrolled." In order to print player/missile displays, one need only specify the address of PMBASE.

While this program is loaded (it only requires 2176 bytes of memory), normal programming and execution may be done, including listing data to the printer with the usual LPRINT or LIST commands, even though the 850 Interface is not connected. Also supplied is an autobooting printer driver program you can load (in very little memory) if you don't want screen dump ability but do want the ability to print data without having to reconnect the 850 Interface.

In summary, Macrotronics has done a FANTASTIC job in making an extremely versatile, easy to use and well documented screen printing program. It will work with or without a running program, display fine scrolled data, display player/missile graphics and, of course, any custom display list you can concoct. On top of that it allows printing data to the printer (e.g. program listings) without use of the 850 Interface. And what's even better is the very reasonable price of this hardware/software package. *I recommend it highly!*

What's GNU (Chapter 2)

- * We will have a demo disk of 5 programs from London Software, including TRION. They have offered our Group a substantial (!) discount on Group orders of 25 or more pieces. Be sure to be at the June meeting.
- * From LAACE (5/83): G.A.M.E.S. will give a 20% discount off retail price on all software for the Atari computer if you present a valid Club membership card (local store: 2814 W. Sepulveda, Torrance)!!!
- * From ACE (Eugene, Ore.) (6/83):
 - rumors on the new line of computers & peripherals (M. Benioff)
 - new software for PCS line from Atari: Pole Position, Joust, Tempest, Battlezone, Pengo, Moon Patrol, Kangaroo, Robotron, Jungle Hunt, Vanguard, Microsoft Basic 2, DOS 3.0, Dig Dug, Assembler Editor 2, New Basic, Eastern Front (ROM), Donkey Kong, Tennis, Logo, Ms. PacMan, Superman 3, Music System 1+2, Liberator. (M. Benioff)
 - Regular monthly sections on PILOT and ASSEMBLY
- * From LAACE (6/83):
 - Atari BASIC Reference Manual Update (from Atari)
 - Review of Data Perfect (Ray Maynard)
- * From ACAOC (5/83):
 - Full-View 80 review (Russell Kavanagh)
 - Description of the NCT 810 Turbo modification for the 810 drive converting it to double density, as demo'd at our last meeting
- * From ACE (Eugene, Ore.) (1/83):
 - article on Burst I/O by Pat Warnshuis
- * From Huntsville Atari Users Group (5/83):
 - Human Engineering in Personal Computers by David Hayes
 - Review of Atari Writer word processor cartridge (by Mike Dunn of ACE of Eugene, Oregon)
- * From ACE (Eugene, Ore.) (9/82):
 - Review of Full-View 80 by David Stellmack of ACE of Columbus
- * From CIO (5/83):
 - Tape Loading Program to help find the correct counter # on the cassette recorder by Dan Chun
 - Many "disk errors (144 & 173) may be avoided by making sure your 810 is powered by a power supply with a "30 VA" rating, as opposed to the "15.3 VA" rating of the computer's
- * Attn: Ham Radio Operators (& all Atari users): we now are receiving Ad Astra, The Atari Microcomputer Network run by Jack McKirgan II (WD8BNG).
- * From Keeping PACE (5/83):
 - a review of Bank Street Writer by Rick Gierl
 - Program Overlays with Atari Basic by Steve Monn of the Frederick Atari Computer Technical Society
- * From Current Notes (5/83):
 - article on layout of memory in the computer
- * From ACE (Eugene, Ore.) (5/83):
 - review of the ATR8000 Interface by Pat Warnshuis of Portland ACE, who also reviewed the Monarch and Data-soft Basic Compilers
 - a very positive review of DISKWIZ written by our own Jerry Allen (congrats!)

Mosaic 64K Ram Select

by Jim Anderson

The Mosaic Ram Select board is an add on memory board for Atari computers that allows expansion to 64K bytes of memory and beyond. This article describes some of the features of the Ram Select board and presents a program that will allow the use of the extra memory as a disk drive. The memory disk drive allows extremely fast access to data and programs although it's contents are lost when the computer is turned off.

Ram Select Features

The Ram Select may be used in either the Atari 400 or 800. A single board may be used in the 400. Up to three Ram Select boards may be used in the 800. This allows up to 192K of memory in an Atari 800. The 800 may also be used with one Ram Select board and two Atari 16 K boards for a 96K configuration or with several different combinations of Ram Select and Mosaic 32K boards.

A single Ram Select board in the 400 or 800 may be used either as a bank switched board (more on this later) or to emulate an Atari 1200. The 1200 emulation mode allows for 62K of ram with a 2K "hole" for I/O beginning at \$D000. The choice of whether to use bank select or 1200 emulation is made at the time the Ram Select is installed. Installation for the 1200 emulation mode requires a single solder connection in the 800. Other than that the installation requires no soldering although the computer must be disassembled to add some cables. The installation took me about a half-hour. The instructions were clear, although there was one picture that showed wires running in the opposite direction from the way they were really supposed to go.

Bank switching is probably the most useful way to configure the Ram Select boards. In this mode the lower 48K of memory is always available as in any other 48K configuration. The additional memory is accessed by 4K banks one of which is active at a time. The switchable banks are addressed from \$C000 to \$CFFF. The active bank is selected by storing into location \$FFC0 + bank number. The data written is unimportant, it is the act of storing into the appropriate address that activates the bank. The bank select locations are part of the OS ROM, so no memory is actually changed. Banks are numbered starting from zero. With a single Ram Select board there are 4 banks numbered zero through three. A 96K configuration has 12 switchable banks and the maximum 192K configuration has 36 switchable banks.

When the Atari is powered up bank zero is selected and the OS ROM will recognize the additional memory for a total of 52K. Some available software, such as word processors may make use of this memory with no alterations. If a cartridge is in place the additional memory will not be recognized by the OS, but may be used by a "clever" BASIC programmer for instance.

The Memory Disk Drive

The program presented here allows use of the Ram Select as a memory disk drive. It is designed to run on OS/A+, but should be easily modifiable to run on DOS II. (OS/A+ and DOS II use the same FMS and the "patch" into FMS to

enable the memory disk is at the same location.) It assumes a configuration of a single Ram Select board and two Atari 16K boards. Modification for a 128K configuration would be fairly trivial. The 196K configuration would require use of 256 byte "sectors" and additional FMS changes. The memory drive is accessed as D2:. The device number is simple to change.

There is also a bug in OS/A+ version 1.2 that must be fixed to avoid losing the memory drive when system reset is hit. The problem is that OS/A+ steps on DOSINI whenever it is entered through CPALOC (OS/A+ owners will know what I am talking about.) The following patches may be applied to version 1.2 to correct the problem. Make the patches and write a new DOS.SYS file before loading the memory disk handler. The patches are made using the assembler/editor or EASMD debugger. The problem has been fixed in version 2.1 of OS/A+.

```
15A0 < RTS
15BD < JMP $1641
1641 < LDA #1
1643 < STA $09
```

The memory drive program consists of two parts, an initialization routine that is used once and then goes away, and the "memory handler" that is kept in memory. The initialization routine "formats" the memory disk (i.e. creates a directory and VTOC), links the memory handler into the DOSINI chain, relocates the memory handler to the current location in MEMLO, makes the necessary patch to FMS to enable the memory disk and then exits through the memory handler's DOSINI routine to set MEMLO to a new value above the memory handler. The memory handler intercepts all FMS calls to the OS ROM's SIO routine. If the call is for the memory disk then the data is transferred to or from the bank memory rather than calling SIO. If the call is not for the memory disk then SIO is called.

The drive used as the memory drive (D2: in this case) must be configured in the DOS so that FMS initialization will allocate the drive buffer. My OS/A+ release disk already had drive D2: configured. The procedure for configuring drives may be found in either the OS/A+ reference manual or the DOS II reference manual. The device number of the memory drive may be changed by changing the value of MDEV in the memory drive program. The corresponding device must then be configured in DOS. The number of memory banks supported may be changed by changing the value of BANKS in the memory drive program. Any number of banks from 12 to 22 may be supported.

The memory drive is installed on OS/A+ by assembling it to a file called MDRIVE.COM and then incanting "MDRIVE" at the D1: prompt. It may be made the default drive by incanting "D2:". MDRIVE may be used in conjunction with the RS232 handler although the system reset fix described by Bill Wilkinson in COMPUTE! should be applied to RS232.OBJ. In fact, it is permissible to use the memory drive while an RS232 port is open in concurrent I/O mode. MDRIVE may be loaded either before or after the RS232 handler.


```

0100 .TITLE "MDRIVE - MEMORY DISK DRIVE"
0110 .TAB 14,20,28

```

```

0120 ;
0130 ; MEMORY DISK HANDLER
0140 ;
0150 ; THIS HANDLER INTERCEPTS THE FMS
0160 ; CALLS TO SIO FOR D2: . CALLS FOR
0170 ; OTHER DEVICES ARE PASSED ON TO
0180 ; SIO. D2: IS THE MEMORY DRIVE.
0190 ;
0200 ;

```

```

0210 ; DCB EQUATES
0220 ;

```

```

=0300 0230 DCB      =      $0300
=0300 0240 DDEV     =      DCB      ; DEVICE NUMBER
=0301 0250 DUNIT    =      DCB+1    ; UNIT NUMBER
=0302 0260 DCMD     =      DCB+2    ; DEVICE COMMAND
=0303 0270 DSTAT    =      DCB+3    ; DEVICE STATUS
=0080 0280 DWRT     =      $80     ; WRITE BIT IN DSTAT
=0040 0290 DREAD    =      $40     ; READ BIT IN DSTAT
=0304 0300 DBUF     =      DCB+4    ; DATA BUFFER ADDRESS
=0308 0310 DLEN     =      DCB+8    ; DATA BUFFER SIZE
=030A 0320 DSEC     =      DCB+10   ; SECTOR NUMBER
0330 ;

```

```

0340 ; FMS EQUATES
0350 ;

```

```

=07A2 0360 FMSSIO   =      $07A2   ; JSR SIOV IN FMS
0370 ;

```

```

0380 ; OS EQUATES
0390 ;

```

```

=0030 0400 SBUFR    =      $30     ; SECTOR POINTER
=0032 0410 DBUFR    =      $32     ; DATA BUFFER POINTER
=0034 0420 ZSCR     =      $34     ; TWO BYTES SCRATCH
=E459 0430 SIOV     =      $E459   ; SIO ENTRY VECTOR
=006A 0440 RAMTOP    =      $6A     ; TOP OF RAM PAGE #
=02E7 0450 MEMLO    =      $02E7
=000C 0460 DOSINI    =      $0C     ; DOS INIT VECTOR
=00D4 0470 FR0      =      $D4     ; USED DURING INIT
=0340 0480 IOCB     =      $0340   ; IOCB 0 ADDRESS
=E456 0490 CIOV     =      $E456   ; CIO ENTRY VECTOR
0500 ;

```

```

0510 ; MISC EQUATES
0520 ;

```

```

=FFC0 0530 BANK     =      $FFC0   ; BANK SELECT BASE
=C000 0540 BNKADR    =      $C000   ; MEMORY BANK ADDRESS
=000C 0550 BANKS     =      12      ; NUMBER OF BANKS
=0180 0560 SECTORS   =      BANKS*32
=0002 0570 MDEV      =      2       ; RAMDISK DEVICE NUMBER
0580 ;

```

```

=2500 0590 ORG      =      $2500
0600 ;
0610 ;

```

```

0620      *=      ORG
0630 ;

```

```

=2500 0640 START     =      *       ; INITIALIZATION
0650 ;

```

```

0660 ; IF RAMTOP INCLUDES THE BANK SELECT
0670 ; AREA WE MUST MOVE IT DOWN TO $C000
0680 ; AND RE-OPEN IOCB 0 TO RELOCATE THE
0690 ; SCREEN MEMORY.
0700 ;

```

2500 A0C1	0710	LDY	##C1	; CHECK RAMTOP
2502 C46A	0720	CPY	RAMTOP	
2504 B013	0730	BCS	I0	; ALREADY OK
2506 88	0740	DEY		; SET Y TO #C0
2507 846A	0750	STY	RAMTOP	
2509 A00B	0760	LDY	##0B	; BYTES TO MOVE TO IOCB 0
=250B	0770 I00	=	*	
250B B98D25	0780	LDA	MYICB,Y	; THIS IS NOT REALLY
250E 994003	0790	STA	IOCB,Y	; LEGAL!
2511 88	0800	DEY		
2512 10F7	0810	BPL	I00	
2514 A200	0820	LDX	#0	; OPEN IOCB 0
2516 2056E4	0830	JSR	CIOV	; ASSUME IT WORKS
=2519	0840 I0	=	*	
	0850 ;			
	0860 ;			CLEAR THE AREA OF THE MEMORY DISK THAT
	0870 ;			CONTAINS THE VTOC AND DIRECTORY.
	0880 ;			
2519 A9C8	0890	LDA	##C4+4	; CLEAR 4 PAGES OF MEMORY
251B 8531	0900	STA	SBUFR+1	; FROM #C400
251D A900	0910	LDA	#0	
251F 8530	0920	STA	SBUFR	; ZAP LO BYTE
2521 A8	0930	TAY		; ZERO TO Y
2522 8DCBFF	0940	STA	BANK+11	; ENABLE VTOC BANK
2525 A205	0950	LDX	#5	; CLEAR 5 PAGES
=2527	0960 I1	=	*	
2527 9130	0970	STA	(SBUFR),Y	
2529 88	0980	DEY		
252A D0FB	0990	BNE	I1	
252C C631	1000	DEC	SBUFR+1	
252E CA	1010	DEX		
252F D0F6	1020	BNE	I1	
2531 E631	1030	INC	SBUFR+1	; POINT TO VTOC
2533 A004	1040	LDY	#4	; SET VTOC CONSTANTS
=2535	1050 I2	=	*	
2535 B98825	1060	LDA	VTCNST,Y	
2538 9130	1070	STA	(SBUFR),Y	
253A 88	1080	DEY		
253B 10F8	1090	BPL	I2	
253D A039	1100	LDY	#[SECTORS/8]+9	; SET BIT MAP
253F A9FF	1110	LDA	##FF	
=2541	1120 I3	=	*	
2541 9130	1130	STA	(SBUFR),Y	
2543 88	1140	DEY		
2544 C00B	1150	CPY	#11	
2546 B0F9	1160	BCS	I3	
2548 A97F	1170	LDA	##7F	; ALLOCATE SECTOR 0
254A 9130	1180	STA	(SBUFR),Y	
254C A038	1190	LDY	#45+11	; ALLOCATE VTOC AND DIRECTORY
254E 9130	1200	STA	(SBUFR),Y	; SECTOR #170
2550 A900	1210	LDA	##00	
2552 88	1220	DEY		
2553 9130	1230	STA	(SBUFR),Y	; SECTORS #168 - #16F
	1240 ;			
2555 A50C	1250	LDA	DOSINI	; SAVE OLD DOSINI
2557 8D0126	1260	STA	HANDLER+1	; SO WE CALL IT
255A A50D	1270	LDA	DOSINI+1	
255C 8D0226	1280	STA	HANDLER+2	
	1290 ;			
255F ADE702	1300	LDA	MEMLO	; MOVE HANDLER DOWN TO MEMLO
2562 85D4	1310	STA	FR0	; USE FR0 FOR (),Y

```

2564 850C      1320      STA      DOSINI ; THIS IS NEW DOSINI
2566 18        1330      CLC
2567 6912      1340      ADC      #MEMIOX ; AND PATCH INTO FMS
2569 8DA307    1350      STA      FMSSIO+1
256C ADE802    1360      LDA      MEMLO+1 ; NOW THE HI BYTE
256F 85D5      1370      STA      FR0+1 ; TO INDIRECT POINTER
2571 850D      1380      STA      DOSINI+1 ; AND INIT VECTOR
2573 6900      1390      ADC      #0 ; AND FINISH THE PATCH
2575 8DA407    1400      STA      FMSSIO+2 ; TO FMS
2578 A07B      1410      LDY      #HND5IZ
      =257A      1420 I4      =      * ; MOVE IT DOWN
257A B90026    1430      LDA      HANDLER,Y
257D 91D4      1440      STA      (FR0),Y
257F 88        1450      DEY
2580 10FB      1460      BPL      I4 ; BETTER BE LESS THAN 128 BYTES
      1470 ;
2582 8DCCFF    1480      STA      BANK+BANKS ; SELECT BOGUS BANK
2585 4C0326    1490      JMP      SMLO ; SET NEW MEMLO
      1500 ;
      1510 ; CONSTANTS FOR VTOC
      1520 ;
2588 00        1530 VTCNST .BYTE $00 ; FMS VERSION
2589 7F01      1540 .WORD SECTORS-1 ; TOTAL SECTORS
258B 7601      1550 .WORD SECTORS-10 ; FREE SECTORS
      1560 ;
      1570 ; THIS IS A DANGEROUS AND ILLEGAL WAY
      1580 ; TO CLOSE AND REOPEN IOCB 0. IT
      1590 ; WORKS AS LONG AS IOCB 0 IS NOT OPEN
      1600 ; TO A DISK FILE.
      1610 ;
258D FF        1620 MYICB .BYTE $FF ; CLOSED
258E 00        1630 .BYTE 0 ; DEV# (DONT CARE)
258F 03        1640 .BYTE 3 ; OPEN COMMAND
2590 00        1650 .BYTE 0 ; STATUS (DONT CARE)
2591 9925      1660 .WORD EDEV ; DEVICE NAME
2593 0000      1670 .WORD 0 ; PUT ROUTINE (DONT CARE)
2595 0200      1680 .WORD 2 ; BUFFER LENGTH
2597 0C        1690 .BYTE $0C ; OPEN INOUT
2598 00        1700 .BYTE 0 ; GRAPHICS 0
      1710 ;
2599 453A      1720 EDEV .BYTE "E:" ; DEVICE NAME
259B           1730 *= START+256
      1740 ;
      1750 ; THE HANDLER IS RELOCATABLE CODE.
      1760 ; THE PORTION BEGINNING AT HANDLER
      1770 ; GETS CALLED THROUGH THE DOS INIT
      1780 ; VECTOR ON EVERY SYSTEM RESET TO
      1790 ; RE-ESTABLISH MEMLO.
      1800 ;
      =2600      1810 HANDLER = *
2600 200000    1820 JSR 0 ; CALL OLD DOSINI
      =2603      1830 SMLO = * ; SET NEW MEMLO
2603 18        1840 CLC
2604 ADE702    1850 LDA MEMLO
2607 697B      1860 ADC #HND5IZ
2609 8DE702    1870 STA MEMLO
260C 9003      1880 BCC SMLOX
260E FEE802    1890 INC MEMLO+1
      =2611      1900 SMLOX = *
2611 60        1910 RTS
      =0012      1920 MEMIOX = *-HANDLER

```



```

1930 ;
1940 ; THE PORTION OF THE HANDLER BEGINNING
1950 ; AT MEMIO RECEIVES FMS CALLS TO SIO.
1960 ;
      =2612
2612 AD0103 1980 MEMIO = *
2615 C902 1990 LDA DUNIT ;GET REQUESTED DEV #
2617 D03F 2000 CMP #MDEV ; IS IT OUR DEVICE
2619 AD0403 2010 BNE JSIO ;NOPE, OFF TO SIO
261C 8532 2020 LDA DBUF ;MOVE BUFFER ADDRESS
261E AD0503 2030 STA DBUFR ;TO ZERO PAGE
2621 8533 2040 LDA DBUF+1 ;SO WE CAN ADDRESS
2623 AD0B03 2050 STA DBUFR+1 ;INDIRECTLY.
2626 8535 2060 LDA DSEC+1 ;MOVE HI OF SECTOR TO ZP
2628 AD0A03 2070 STA ZSCR+1
262B 4635 2080 LDA DSEC ;GET LO OF SECTOR NUMBER
262D 6A 2090 LSR ZSCR+1 ;GET LO OF SEC ADR
262E 48 2100 ROR A
262F A900 2110 PHA ;SAVE THIS FOR A SEC
2631 6A 2120 LDA #0
2632 8530 2130 ROR A ;LO OF MEMORY ADR TO A
2634 68 2140 STA SBUFR ;LO OF SECTOR POINTER
2635 48 2150 PLA ; GET HI OF ADR
2636 290F 2160 PHA ; SAVE IT AGAIN
2638 09C0 2170 AND #00F ;LO NIBBLE IS OFFSET
263A 8531 2180 ORA #C0 ;BEGINNING OF BANK
263C 68 2190 STA SBUFR+1 ;HI OF SECTOR ADDRESS
263D A204 2200 PLA ; GET HI OF ADR AGAIN
      =263F 2210 L1 = *
263F 4635 2220 LDX #4 LOOP COUNT
2641 6A 2230 LSR ZSCR+1 ; CALCULATE BANK NUMBER
2642 CA 2240 ROR A
2643 D0FA 2250 DEX
2645 AA 2260 BNE L1
2646 8634 2270 TAX ;BANK NUMBER TO X
2648 9DC0FF 2280 STX ZSCR ;DEBUG
264B A07F 2290 STA BANK,X ;ENABLE BANK
264D 2C0303 2300 LDY #7F ;BYTE COUNT TO Y
2650 3020 2310 BIT DSTAT ;READ OR WRITE?
2652 7007 2320 BMI WRITE
      2330 ;
      =2654 2340 BADCMD = * ;UNKNOWN COMMAND
2654 A98B 2350 LDA #8B ;DEVICE NAK ERROR
2656 D013 2360 BNE XIT ;AND DONE
      2370 ;
2658 4C59E4 2380 JSIO JMP SIOV ;GO OFF TO SIO
      2390 ;
      =265B 2400 READ = *
265B AD0203 2410 LDA DCMD ;MAKE SURE ITS A READ
265E C952 2420 CMP #52
2660 D0F2 2430 BNE BADCMD ;NOPE, ERROR
      =2662 2440 R1 = *
2662 B130 2450 LDA (SBUFR),Y MOVE DATA
2664 9132 2460 STA (DBUFR),Y
2666 88 2470 DEY
2667 10F9 2480 BPL R1
      =2669 2490 GOODX = *
2669 A901 2500 LDA #01 ;GOOD STATUS
      =266B 2510 XIT = *
266B 8D0303 2520 STA DSTAT ;RETURN STATUS
      2530 ;

```

```

2540 ; WE NEED TO SELECT A NON-EXISTENT
2550 ; BANK BECAUSE OF THE BUG IN THE SCREEN
2560 ; CLEAR ROUTINE THAT CLOBBERS MEMORY
2570 ; BEYOND THE END OF THE SCREEN.
2580 ;
266E 8DCCFF      STA   BANK+BANKS
2671 60          RTS
2610 ;
=2672      2620 WRITE      =      *
2672 B132      2630      LDA   (DBUFR),Y ;MOVE TO SECTOR
2674 9130      2640      STA   (SBUFR),Y ;
2676 88        2650      DEY
2677 10F9      2660      BPL   WRITE
2679 30EE      2670      BMI   GOODX ;AND DONE
=007B      2680 ;
2690 HNDISZ    =      *-HANDLER
2700 ;
267B      2710      .END

```

EPSON MX-80 AND THE ATARI WORD PROCESSOR

by A.Koudounaris (Phone 213 374-4871)

I purchased the Atari word processor (Version 1.0) and the Epson printer (MX-80 FT with Graftrax plus) at the same time. Learning the word processor took me awhile since I was entirely unfamiliar with word processing in general. As to the Epson, the book of instructions is excellent, however, it can be somewhat frustrating when it comes to applying it to the word processor. After considerable experimentation I prepared a table for my convenience that I believe will help those of you who have the same hardware and software. I keep a copy of this table handy when I am typing.

The surest way to get the Epson to do what you want it to do is to feed it string characters (CHR\$(123)), but they are a pain to type; besides, the word processor will not allow you to enter strings. Control and escape characters are more convenient to use and they will work.

The table below shows some of the most useful functions that the Epson can print. One column shows how to get those functions using Basic or direct mode. In this case you use the LPRINT or PRINT "P:" instruction and type quotation marks to enclose the special characters. The ESCAPE key is pressed twice. For the word processor, insertion of the characters is done while in the Edit mode by typing CONTROL INSERT then entering one special character; if a second special character is required then CONTROL INSERT must be entered again. A special character example is ESCAPE G, in this case the ESCAPE key is pressed once only.

Note that in some cases using lower case letters in an instruction will not work. That is why in the Table the word CAPS is used.

Two recent articles have appeared on this subject and may be of help also; one is in Analog, issue number 10, p.61 "Epson Printing Modes Simplified". The other is in Compute!, April '83 "Using the Atari Word Processor with an Epson Printer", p.157.

EPSON SPECIAL CHARACTER FUNCTIONS

FUNCTION	ON/ OFF	BASIC OR DIRECT MODE	WORD PROCESSOR CI=CTRL/INSERT
<u>UNDERLINED</u>	ON	"ESC/ESC - CTRL/A"	CI:ESC,- CI:CTRL/A
	OFF	"ESC/ESC - CTRL/,,"	" , CTRL/,
DOUBLE STRIKE (Bold)	ON	"ESC/ESC G"	CI:ESC,G (CAPS)
	OFF	"ESC/ESC H"	CI:ESC,H
EMPHASIZED	ON	"ESC/ESC E"	CI:ESC,E (CAPS)
	OFF	"ESC/ESC F"	CI:ESC,F
ITALICS	ON	"ESC/ESC 4"	CI:ESC,4
	OFF	"ESC/ESC 5"	CI:ESC,5
DOUBLE WIDTH 1 LINE	ON	"CTRL/N"	CI:CTRL/N
	OFF	Not required	Not required
DOUBLE WIDTH CONTINUOUS	ON	"ESC/ESC W CTRL/A"	CI:ESC,W CI:CTRL/A
	OFF	"ESC/ESC W CTRL/,,"	" CTRL/,
MASTER RESET	OFF	"ESC/ESC @"	CI:ESC,@
SUBSCRIPTS	ON	"ESC/ESC S CTRL/A"	CI:ESC,S CI:CTRL/A
SUPERSCRIPT	ON	" " CTRL/,,"	" CTRL/,
SUB/SUPERSC	OFF	"ESC/ESC T"	CI:ESC,T
NOTE:Leaves printer in double strike mode-use master reset to delete			
COMPRESSED	ON	"CTRL/O"	CI:(ATARI)CTRL/O
	OFF	"CTRL/R"	CI:CTRL/R
BUZZER	ON	"ESC/ESC CTRL/G"	CI:ESC.CI:CTRL/G

Notes on getting compressed characters with the word processor.

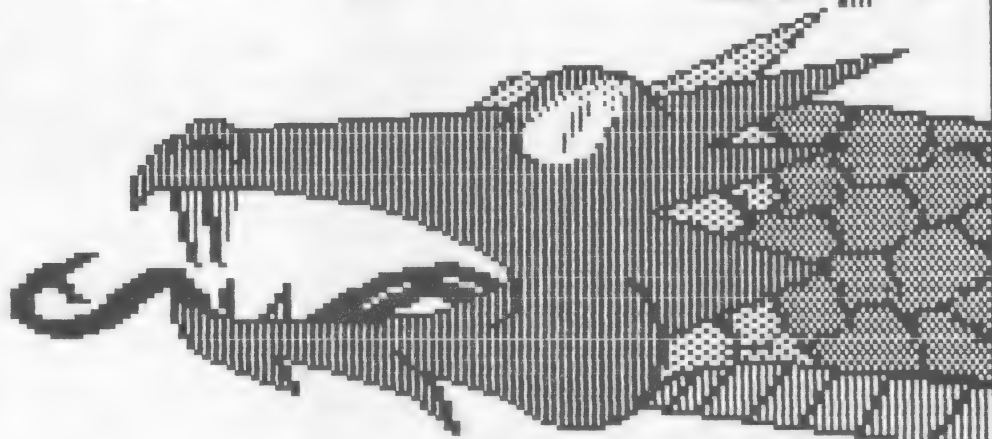
To start printing compressed characters you must use (Atari)control O; either capital or lower case . The catch is that you have to write this entry on the line preceding the one you want to have compressed. To end compressed characters you have to enter control R (or r). Use of 10 or 16 cpi does not have any effect.

Example:

I will place the Atari ctrl O here "X"

This writing is all compressed.When I am finished with compressed, I will then enter the control R,here "X" Which then brings me back to normal.

The Dragon's Eye



Dragon by Brian E. Cagle & Me

CAGLE'S CORNER

Game Review By William Cagle

Hear Ye, Hear Ye, all who would go forth and due battle for the Dragon's Eye. Thou wilt need an Atari 400/800 computer with a memory of 40K and the Basic cartridge.

To begin, thou will be asked thy name, then thou shall be asked to pick a title and the weapon thou shall use. At this point thou shall be given a scroll of spells to use during the adventure. This entire adventure shall be controlled by the use of the Keyboard.

Now the quest begins. There is a time limit of twentyone (21) days by the game calandar so watch it. You are in a part of a province called the Fel City. Remember well where you start so you may return there at the end of the quest. In all there are six (6) other provinces in the Kingdom and somehow you must find the Eye in one of them. The other provinces are the Lofty Mountains, the East Plain, the Deep Chasm, Ley Land, the West Plain, and the Dark Forest. Use well your spells and weapons, for there are many monsters to be fought, from bats to dragons. There are also many treasures to be found along with the Eye.

I myself donot do well with adventure games, but this one by EPYX was different. It's graphics and sound were very good and I could keep track of what I was doing. So on the Cagle scale of games I give this one a 7+.

Try this game, I think you'll be pleasantly surprised.

." FORTH & BACK "

By D.J. Whitehead

Well, here we are again, it is time for me to set down at the TSO (for you forthers, =>

; TSO ." Two Seater Outhouse " ;)

This issue I am hitting the high points of FORTH and Atari Graphics. Those of you who attended the FEB. SBACE meeting followed by the Forth users get together will recall a challenge to improve a simple graphics screen. This set of two screens, along with a simple set of explanations will make up this issue's column.

SCR # 20

```
0 ( GRAPHICS EXT-8/82 SCR 50 REQ'D)
1 ( 88 @ -> SCREEN 0,0 LOCATION )
2 ( 88 @ 1 + C@ -> 1,0 SCR CONTS)
3
4 ; R/WSCR 1 + 0 DO 88 @ I 128 * + ( ADDR)
5 I PAD 2+ @ + PAD @ ( ADDR B# F-)
6   1 SWAP -DISK DROP DROP LOOP ;
7 ; GSCR 5 GR 25 R/WSCR ;
8 ; STO 600 PAD 2+ ! 0 PAD ! ;
9 ; RCL 600 PAD 2+ ! 1 PAD ! ;
10
11
12
13
14
15
```

SCR # 21

```
0 ( JOYSTICK DRAWING REQ' SCR 20)
1 1 VARIABLE CURS
2 40 VARIABLE GX 20 VARIABLE GY
3 0 VARIABLE DGX 0 VARIABLE DGY
4 ; GLOC 0 0 DGX ! DGY ! 632 C@
5   DUP 8 AND 0= IF 1 DGX ! ENDIF
6   DUP 4 AND 0= IF -1 DGX ! ENDIF
7   DUP 2 AND 0= IF 1 DGY ! ENDIF
8   DUP 1 AND 0= IF -1 DGY ! ENDIF
9   DROP ;
10 ; GTRI 644 C@ 0= IF 0 CURS !
11   ELSE 1 CURS ! ENDIF ;
12 ; GLIM GX @ DGX @ + 0 MAX 79 MIN GX !
13   GY @ DGY @ + 0 MAX 79 MIN GY ! ;
14 ; DRW 5 GR BEGIN GLOC GTRI GLIM CURS @
15 GX @ GY @ PLOT ?TERMINAL UNTIL ;
```

The above screens require that screen 50 from the SBACE fig FORTH 1.1s is loaded. This can be performed by

50 LOAD <cr>

Then load the above screen 20 followed by screen 21. A brief explanation of each of the words in screen 20, and 21 follows..

The first three lines of scr 20 are comments. Note line 0 of any screen will be displayed by the use of the FORTH command `index`. Lines 2 and 3 are reminders that loc 88 decimal contains the address of the CRT. Therefore by looking at the CONTENTS of memory location 88 base ten, you can see what is at $x=1, y=1$ on the screen. (Note in forth $x=1$ is $1\ x!$ for a 16 bit word, or $1\ x c!$ for a 8 bit word.) Scr 20 is used for the storage and recall of the graphics image. The user is only interested in three words defined on this screen, these are ...

GSCR -> used to specify a Graphic SCReen command

STO -> specify a STORage function

RCL -> specify a ReCaLl function

(Yes, I am from the HP calculator school thus STO and RCL words are used a lot.)

The proper syntax is ...

RCL GSCR (-> displays the graphics information stored at disk block 600 to 625.)

STO GSCR (-> saves the graphics information at disk blocks 600 to 625.)

The word R/WGSCR is the actual commands to operate the disk drive. (It should be pointed out that a graphics 5 screen does not require all 25 disk blocks. I am leaving it to the readers to find out the smallest number of disk blocks to use.) Note by separating them out, STO and RCL become simple words to define.

Lets examine screen 21 in detail. First line 0 is a comment for INDEX to pickup. Note that screen 20 is indicated as required. This is in part due to my having generated a neat picture for my son, using just screen 21, and then being unable to save it!!! Thus for your own sanity it is best to have at least a crude method of saving your work available. Note lines 1 to 3 define 5 variables in FORTH. For example, 1 variable `curs` in FORTH is equivalent to writing `curs = 1` in basic. In general CURS is the curser state (1-> on/ 0-> off). GX and GY are the Graphics X and Y location of the curser, while DGX and DGY are the Delta, or offset from the current GX and GY which the user has commanded by pushing the joystick. Review of SCR 21 shows the default (starting) values for the X and Y location to be 40 and 20 units. This is about the center of a graphics 5 Atari screen. The word GLOC is defined to read the joystick (plugged into port 0) and determine the correct values for DGX and DGY. The first thing GLOC does is to zero out any prior values in DGX and DGY. This is accomplished by placing zero on the stack twice, then storing the top of the stack first in DGX by the `!` word, then in DGY. For simplicity both DGX and DGY are considered as 16 bit words. However, when memory location 632 decimal is examined, it must be considered as a BYTE (8 bits). This is accomplished by using the word `C@` instead of just the word `@`. For those of you who know the Atari memory map, location 632 is well known as the 0 port joystick location. If the joystick was pushed to the right, the most significant bit (MSB) bit 7 will be set to a 1. When the stick is pushed to the left, then bit 6 is set to a 1. (If both bits 7 and 6 are set, then you have a bad joystick, or you have been playing too much Star Raiders.) By noting that if bit 0 is set the resulting value of the byte is 1, while bit 1 set is a 2, bit 3 set yields a 4, and finally if bit 4 is set an 8 results, allows a simple method to check the joy stick direction. remember to keep track of the stack during this process. In this case I am duplicating the top of the stack prior to each 'AND' command. Thus at the conclusion of the command word GLOC I must clear the STACK by using the 'DROP' word in line 9.

Line 10 contains the definition of the word 'GTRI'. For those of you who use basic peek, and poke, location 644 will be remembered as the location of joystick 0 trigger status. Upon examination of line 10, note I am placing the contents of the trigger byte on top of the stack. The top of the stack is then checked to see if it is or is not zero, by using the '0=' word. The remainder of line 10 and all of line 11 is used to

place a 0 in curs if the trigger button was pressed, or else a value of 1 is assigned to the variable curs.

Lines 12 and 13 define a word called 'GLIM', which is used to define the limits of the graphics area. This word also adds in the offset or movement which the user wanted when he (or she) pressed the joystick. Because this word performs two functions, lets look at it in some detail. In order to make this a little clearer, and to show a method I use when generating new words, lets write the word vertically and place comments beside it.

```

:      start of a colon definition
GLIM  name of definition ( Graphics LIMit )
GX    address of where graphics curser is in the x direction
@     place x value on stack
DGX   address of where the delta x desired is stored
@     place delta x value on stack
+     add the top two items on the stack, placing the result on top.
0     place zero on top of the stack
MAX   compare the top two values of the stack, and keep the larger one.
79    place 79 on top of the stack
MIN   keep the smaller of the top two values of the stack
GX    the address of the ( now updated ) X position.

!     store the second value of the stack, at the address found at the top of the
stack
```

etc for the y dimension in line 13

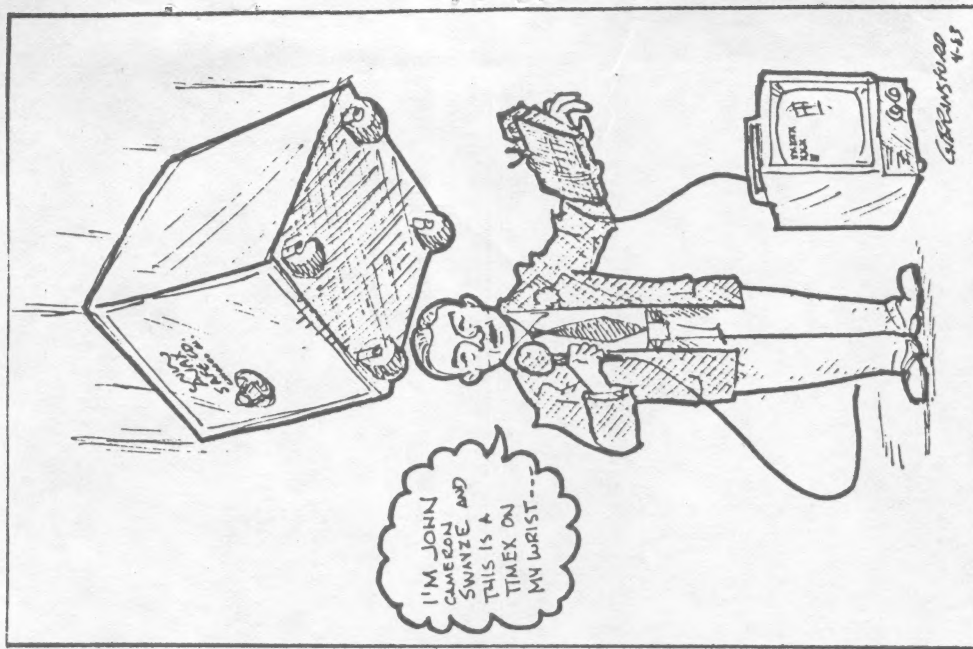
```

;      close out colon definition
```

```

:      start of colon definition
DRW   name of definition for DRaWing on the screen
5     place the value 5 on the stack
GR    the FORTH command ,taking the top number of the stack
      and using it like the basic graphics command GR.
BEGIN  the start of a loop
  GLOC  find out where user wants the curser
  GTRI  does user want the curser on or off?
  GLIM  besure the curser stays on the screen
  CURS @ place status of curser on stack
  GX @  place x location on stack
  GX @  place y location on stack
  PLOT  similar to BASIC, however a third value used is the color register
  ?TERMINAL if a key on keyboard has been pressed, leave a TRUE flag on stack
        else leave a false flag on the stack
UNTIL  pick up flag from stack, if false
      loop back to begin, else fall out of loop
;      terminate colon definition
```

This concludes the explanation of the screens 20 and 21. To use then just load screens and type DRW. Then use a joystick (in port 0) and try your hand at drawing. When you are finished, press any key, and type STO GSCR to save it. remember the challenge still stands, generate improvements to these screens, and I will place them in the column, with YOUR byline.



Editorialette
by James A. Jengo

Dear Software Publisher:

Please don't announce release dates for upcoming programs and software packages if you can't produce! This seems to be common practice, and is very annoying to us program users. I guess you want to get us all excited about your product so we'll want to dash out and buy it as soon as it is released, but what happens is that we quickly get disgusted with all the delays in release and frequently will buy a reasonable substitute when it becomes available. Several companies already have a reputation for this practice. Please have a little more consideration for us. Thank you.

Among other articles and commentaries the next issue will have a review of the new Monkey Wrench II by Eastern House Software (I just received my upgrade yesterday).

Please continue to submit articles, programs and reviews.

Congrats to Robert Smith on his new position!

S. B. A. C. E.

5025 Range Horse Lane
Rolling Hills Est., CA 90274

